Simulate an IoT SmartHome node

Description:

This was a build to control a DC motor that should turn based on the alerts sent to the XBee connected to my computer. This build contains a photo resister, a 1KΩ resister, an H bridge, a DC motor, an Arduino with a sparkfun XBee shield on it, two Xbee's, an XBee explorer and multiple wires.

. Issues:

I had two main issues with this build with the first being getting the motor functioning. I set up the motor as I had in Assignment 3, but for some reason, it wasn't turning at all. Worst, sometimes it would turn in only one direction and then stop working entirely. I was changing so many variables each time I tried to troubleshoot that I could never figure out what was going on. The problem turned out to be the speed I had it set at, which was surprising because in Assignment 3, the code I based my project on had mapped the second potentiometer from 0 to 50, so I thought 50 was a speed the motor could respond to. From this, I learned that example code can be wrong and that messing with the variables yourself is the only way to figure out if something works. Also, when troubleshooting I should only change one variable at a time.

The other issue I ran into was trying to extract the readings the Arduino Xbee was sending to my computer XBee. While the assignment just required having our system respond to alerts sent by XCTU, I wanted to have an independent program that was taking the readings from the second XBee attached to my computer and then would produce and send my specified alerts based on the readings it received. But I couldn't figure out how to get the information from the XBee. While I could see the frames in XCTU, I couldn't figure out how to input that data into another program or bare minimum how to save it. This portion of the assignment made me realize that connecting the devices isn't enough if you can't use the information you are reading from the other device to do anything.

Diagram of Build

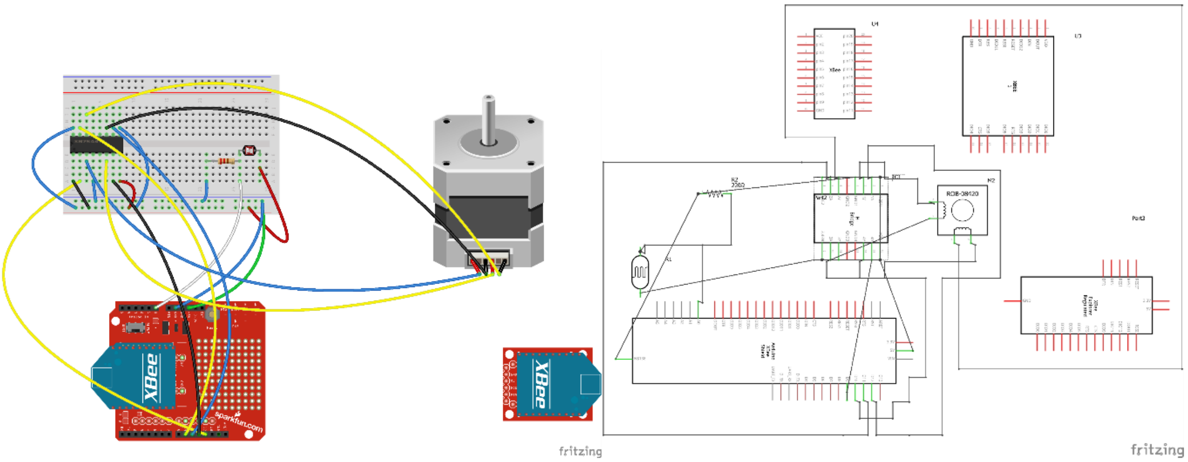
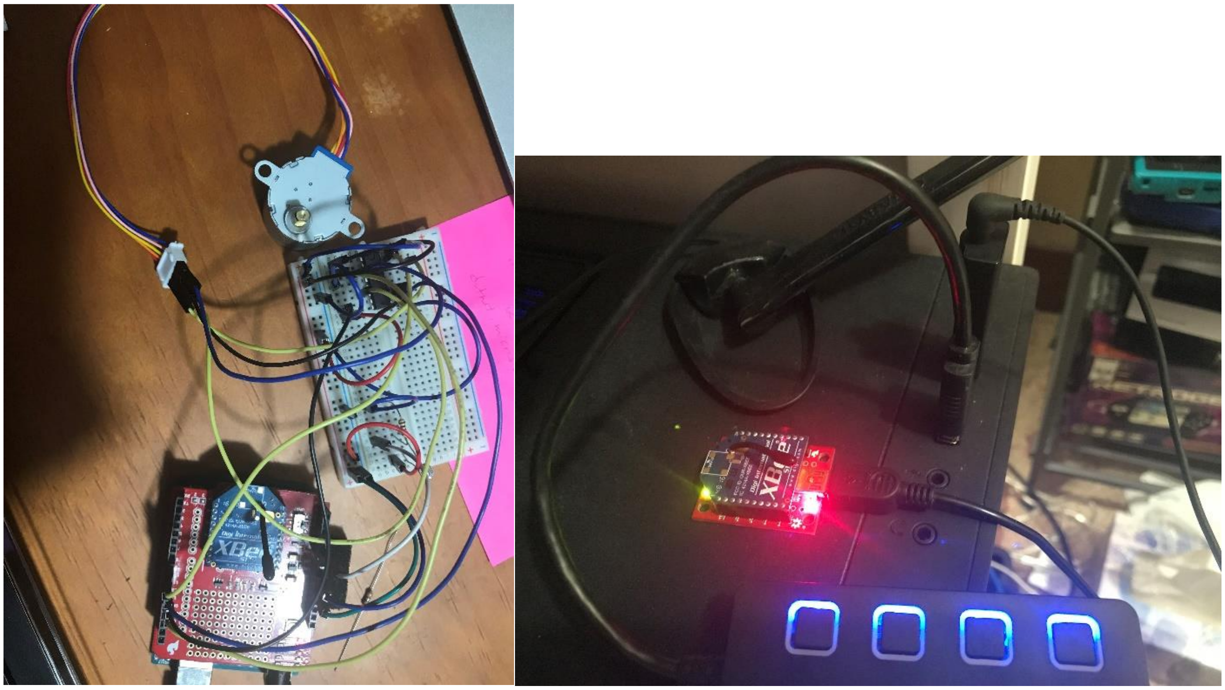


Photo of Build



Code for Build

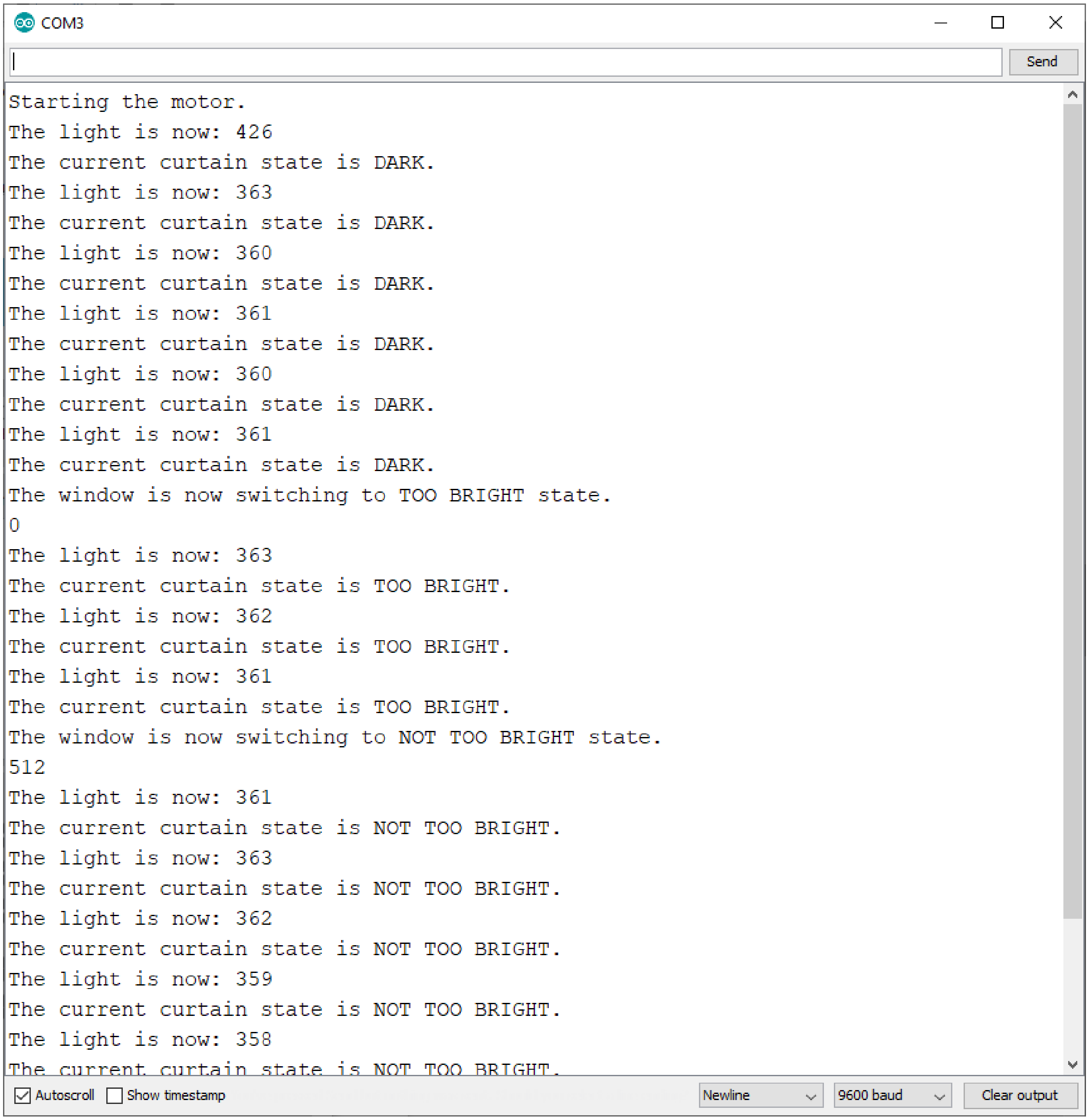
```
/*

Code from https://learn.adafruit.com/adafruit-arduino-lesson-16-stepper-motors/arduino-code.

Includes code from https://learn.sparkfun.com/tutorials/xbee-shield-hookup-guide edited to work with input from a photo resistor and sending output to the motor that is connected to the XBee shield.

Most of the code was pulled from Assignment 3. The idea was the use the motor as a thing that could control a window blind.

*/

#include <Stepper.h>

#include <SoftwareSerial.h>


SoftwareSerial xbee(2, 3);


const int stepsPerRevolution = 512; //number of steps per revolution for the motor


const String statuses[] = {"DARK", "NOT TOO BRIGHT", "TOO BRIGHT"};
```

```
int in1Pin = 12;
int in2Pin = 11;
int in3Pin = 10;
int in4Pin = 9;
int photoPin = 0; //used to read from the photosensor to figure out how to edit the curtain position.
int currentCurtainState = 0; //trying to keep track of what position the curtain is.

int speed = 30;
int steps = 0;
boolean reverse = true;

String previous_status = statuses[0];
String status = statuses[0];

Stepper motor(stepsPerRevolution, in1Pin, in2Pin, in3Pin, in4Pin);

void setup()
{
  pinMode(in1Pin, OUTPUT);
  pinMode(in2Pin, OUTPUT);
  pinMode(in3Pin, OUTPUT);
  pinMode(in4Pin, OUTPUT);
  pinMode(photoPin, INPUT);

  xbee.begin(9600);
  Serial.begin(9600);
  Serial.println("Starting the motor.");
}
```

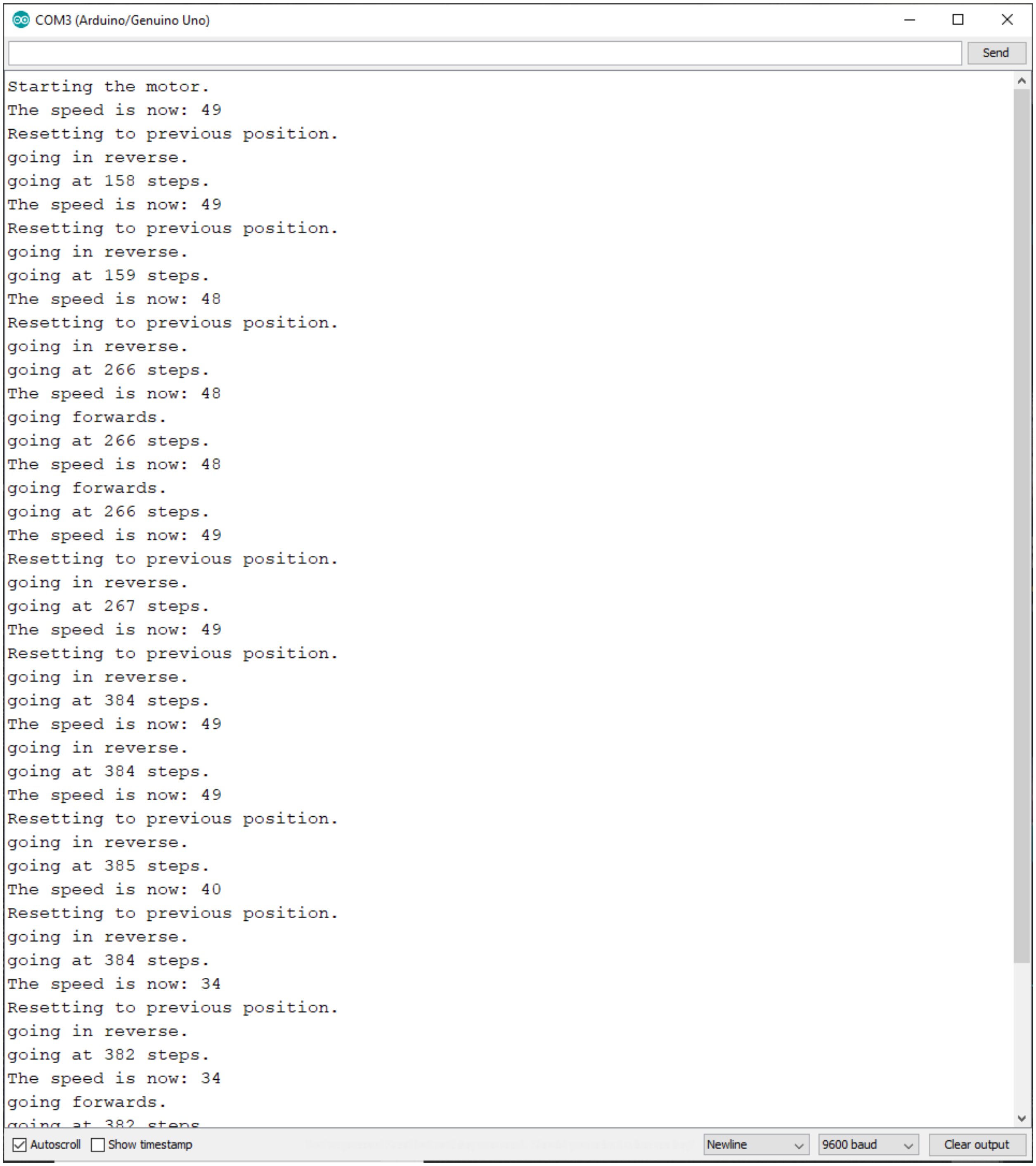
```
void loop()
{
  Serial.print("The light is now: ");
  int lightLevel = analogRead(photoPin);
  Serial.println(lightLevel);
  Serial.print("The current curtain state is ");
  Serial.print(status);
  Serial.println(".");
  xbee.write(lightLevel);

  if(xbee.available()) { //trying to get the arduino to read from the
xbee
    status = String(xbee.readString());
  }

  if(status != previous_status) {
    Serial.print("The window is now switching to ");
    Serial.print(status);
    Serial.println (" state.");
    adjustBlinds(status);
    previous_status = status;
    Serial.println(currentCurtainState);
  }

  delay(10000); /* chose 10 minutes as when we would check it because I
thought any more
  frequently would lead to a constantly shuttering blind effect. */
}

void adjustBlinds(String status)
```

```
{
  motor.setSpeed(speed);
  if(status == statuses[1]) {
       if(currentCurtainState == 256) {
          currentCurtainState+= 256;
          motor.step(256);
       } else {
          currentCurtainState+=512;
          motor.step(512);
       }
  } else if(status == statuses[2]) {
    if(currentCurtainState > 256) {
      currentCurtainState -=256;
      motor.step(-1 * 256);
    }
  } else if (status == statuses[0]) {
    if(currentCurtainState > 256) {
      currentCurtainState -= 512;
      motor.step(-1 * 512);
    } else {
      currentCurtainState -= 256;
      motor.step(-1 * 256);
    }
  } else {
    Serial.println("The input entered is unsupported. Please try
again.");
  }
}
```

Example of Output



```
Starting the motor.
The light is now: 426
The current curtain state is DARK.
The light is now: 363
The current curtain state is DARK.
The light is now: 360
The current curtain state is DARK.
The light is now: 361
The current curtain state is DARK.
The light is now: 360
The current curtain state is DARK.
The light is now: 361
The current curtain state is DARK.
The window is now switching to TOO BRIGHT state.
0
The light is now: 363
The current curtain state is TOO BRIGHT.
The light is now: 362
The current curtain state is TOO BRIGHT.
The light is now: 361
The current curtain state is TOO BRIGHT.
The window is now switching to NOT TOO BRIGHT state.
512
The light is now: 361
The current curtain state is NOT TOO BRIGHT.
The light is now: 363
The current curtain state is NOT TOO BRIGHT.
The light is now: 362
The current curtain state is NOT TOO BRIGHT.
The light is now: 359
The current curtain state is NOT TOO BRIGHT.
The light is now: 358
The current curtain state is NOT TOO BRIGHT.
```



```
Starting the motor.
The speed is now: 49
Resetting to previous position.
going in reverse.
going at 158 steps.
The speed is now: 49
Resetting to previous position.
going in reverse.
going at 159 steps.
The speed is now: 48
Resetting to previous position.
going in reverse.
going at 266 steps.
The speed is now: 48
going forwards.
going at 266 steps.
The speed is now: 48
going forwards.
going at 266 steps.
The speed is now: 49
Resetting to previous position.
going in reverse.
going at 267 steps.
The speed is now: 49
Resetting to previous position.
going in reverse.
going at 384 steps.
The speed is now: 49
going in reverse.
going at 384 steps.
The speed is now: 49
Resetting to previous position.
going in reverse.
going at 385 steps.
The speed is now: 40
Resetting to previous position.
going in reverse.
going at 384 steps.
The speed is now: 34
Resetting to previous position.
going in reverse.
going at 382 steps.
The speed is now: 34
going forwards.
going at 382 steps
```

☑ Autoscroll ☐ Show timestamp Newline 9600 baud